AMENDMENT TO THE SPECIFICATION

Page 2-- amend the first paragraph

[0003] FIG. 1-1 shows a typical database system DBS comprising a primary memory D, e.g. a primary disc memory D, and a data processing device DPD. In the primary memory D data of a database DB is stored as a plurality of data blocks p00, p01, . . . p32. Typically the individual data blocks p are called pages of the database DB, i.e. the data in the primary memory device D is organized as a plurality of pages P. As is indicated in FIG. 1-1, each datablock or data page P consists of one or more data objects OB. As is shown for the datablock P02, P32, each of these pages comprises a plurality of N and M of data objects OB02-1 . . . [[OB 02-M]] OB02-N, OB32-1 . . . OB32-M.

Page 4-- amend the paragraph that carries over to page 5

[0011] When the page identity p02, p32 has been determined in step S1, the next step is to make a lookup in a data structure called the [[has-table]] hash-table HT to identify where the desired pages are currently stored in the main memory, more precisely in the page cache section PCS. Two scenarios can happen, i.e. either the page is already stored in the page cache memory section or the relevant page has not been stored in the page cache memory section. If the page ~~which~~ that contains the desired data object has already been stored, e.g. if the page P02 containing the desired object OB02-2 has already been stored in the page cache memory, then it is only necessary to read out in step S2 the address location AD-P02 of the relevant page P02 from the hash-table HT. In step S3 the processing means PM gives a read-access request to the main memory to read out the data object from the page [[P92]] P02 at the particular memory location AD-P02.

Page 5-- amend the first full paragraph

[0012] Similarly, if the page has not been stored in the page cache memory section PCS, then the processing means PM first calculates the relevant page p32, a load request for loading the relevant page P32 from the database DB is given in step [[S1]] S1'; and then the steps S2, S3 are

repeated with the relevant address AD-P32 of the page where the desired object OB32-2 to be read resides. The data structure used for determining the addresses for the pages is often a hash-based data-structure as explained above.

Page 5-- amend the paragraph that carries over to page 6

[0014] Furthermore, in a particular point in time, pages p10, p01, p02, p32, [[p05]] p00, p22 may have been stored in the page cache as shown in FIG. 1-1, however, frequent read accesses have only been performed to the pages p02, [[p06]] p00 (this is indicated with a hatching from the left bottom corner to the right top corner in FIG. 1-1) whilst pages p01, p22 have only been accessed moderately (indicated with a hatching from the top left corner to the right bottom corner in FIG. 1-1). The page p10 (having no hatching) has not been accessed very frequently. Hereinafter, a page data region PCSP ~~which~~ that has been accessed frequently is also called a "hot" page. Likewise, a page data region PCSP ~~which~~ that is not accessed frequently is also called a "cold" page. Pages having read accesses therebetween are called "warm" pages. As can be seen from FIG. 1-1, due to the fact that always complete pages need to be stored in the page cache, a lot of memory space is occupied in the page cache memory section PCS even though the pages themselves have different read access frequencies because individual data objects can only be accessed by first storing the complete page in the page cache.

Page 10-- amend the first full paragraph

[0024] According to the present invention the read or write or read/write access frequency is determined on a data object level. Therefore, it can be decided when a data object should be moved to a "hotter" data storage region of the secondary memory. Extremely "hot" data objects can thus be moved to an often used data storage section of the secondary memory. Very "hot" data objects can be moved to the main memory for access by the processing means PM and "hot" objects are moved to "hot" pages in the page cache memory. Thus, "hot" data objects can be collected together (in separate pages) in a "hot" memory region and "cold" data does not stay longer in the page cache than needed. It should be noted that this concept is applicable to database systems DBS only using the page cache memory section or using the page cache memory section as well as the resident data section. Therefore, there is not so ~~must~~ much

wasting of main memory space for not so hot objects. This idea to move objects between pages can ensure that hot objects reside on the same pages and thus less memory is wasted for cold objects.

Page 18--amend the first paragraph

[0054] The primary advantage of calculating the "heat" via the read/write access frequency on a data object level is that data objects of comparable relevance for the processing means PM can be collected in the same data storage ~~region~~ section PCS-j, RDS-i. Thus, the processing means PM can overwrite data regions of data storage sections having a lower rank earlier than data regions of data storage sections having a higher rank. Therefore, the main memory MM is not ~~overduely~~ over duly occupied by data (data objects) ~~which~~ that are not frequently used and therefore it is justified--if indeed such data objects are needed--to retrieve them from the disc memory D if required (by retrieving the corresponding page as described with reference to FIG. 1-1).

Page 18--amend the paragraph that carries over to page 19

[0056] It should also be noted that the above described principle is not restricted to the usage of two different memory sections in the main memory MM (i.e. a page cache memory and a resident data work memory section). That is, even if--as explained above--only the page cache memory is used, different data storage regions only for this single memory can be provided with different heat levels to allow a movement of data objects when their respectively calculated heat changes. Therefore, also in this case it can happen that at a particular point in time a data object becomes very relevant due to [[it's]] its high read/write or ~~read/write~~ read or write access frequency. At another time, the same data object may ~~loose it's~~ lose its importance (as indicated by the reduced access frequency) and may actually migrate to a lower order data storage section.

Page 19--amend the paragraph that carries over to page 20

[0059] FIG. 3 and FIG. 4 show a first embodiment of the invention when using two different memory sections in the main memory MM, i.e. a page cache memory and a resident data memory. As seen in FIG. 3, each data storage ~~region~~ section has preferably assigned to it a

predetermined access frequency range pch-1, pch-2; rdh-1, rdh-2. For illustration purposes only two data storage sections are shown. Each access frequency range has an upper and a lower access frequency threshold value pch-1_{low}, pch-1_{up}, pch-2_{low}, pch-2_{up}; rdh-1_{low}, rdh-1_{up}, rdh-2_{low}, rdh-2_{up}.

Page 20--amend the first full paragraph

[0060] In general provisions are made such that the access frequencies of an access frequency range of an (i+1)-th data storage section PCS-i; RDS-i are greater than the read access frequencies of an i-th data storage section and each access frequency range comprises an upper ~~and an~~ and a lower access frequency threshold value, wherein said read/write means R/W is adapted to move a data object of the i-th data storage section from the i-th to the (i+1)-th data storage section when the access frequency of said data object is greater than said upper access frequency threshold value and/or to move a data object of the (i+1)-th data storage section from the (i+1)-th to the i-th data storage section when the access frequency of said data object is smaller than said lower access frequency threshold value.

Page 21--amend the second paragraph

[0065] Firstly, the object OB* can be moved up to the data storage section [[PCS-2]] RDS-3. Furthermore, it can be moved up to the data storage section PCS-3. Furthermore, it can be moved [[up]] over to the data storage section [[RDS-3]] PCS-2. Furthermore, it can be moved down to the data storage section PCS-1 or it can be moved down to the data storage section RDS-1. This is indicated with possibilities P1, P2, P3, P4, P5 for the object OB* in FIG. 3. If it has been determined in step ST5 (on the basis of comparing the respective data object access count with the corresponding threshold) that the object should be moved, then the respective moving of the object takes place in step ST6 depending on the determined access frequency.

Page 24--delete the second full paragraph and the paragraph that carries over to page 25 and add two new paragraphs after the first full paragraph (line 17)

FIG. 5a shows a situation where a user requests access to data belonging to a particular customer. The user inputs a particular so called logical reference "c_ID1", for example a customer identification or a customer name. This is a logical reference key. In the processing means PM, an index INX (i.e. a data structure reference table) must be provided in order to map this logical reference to a physical reference (page reference and page index) pointing to the page section where a data object OB belonging to "c_ID1" may be located. Note that the physical reference p_ID1 can equally refer to the position in the disc database DB or in the main memory sections, for example in the page cache section PCSP.

If, according to the invention, the data object would be moved from p_ID1 to p_ID* (i.e. to a different heat level), e.g. in the page cache section PCSP as indicated with the movement "M", then of course the usage of the physical reference p_ID1 is not correct any longer for locating the respective data object. This becomes even worse if the data object is stored back not at the original position p_ID1 in the disc database DB (see ST2) but at the corresponding page p_ID* in the disc database DB (see ST1). Then the physical location that is assigned to the logical reference in the index structure INX is incorrect, i.e. it does not point to the actual physical location where the moved data object OB is now stored. Therefore, according to the invention, the processing means PM comprises a reference updating means REF-UP that receives the new physical reference p_ID* and updates the corresponding entry in the index structure INX. If now the same user or a further user attempts to access data belonging to the logical reference c_ID1, then the new physical reference p_ID* will be indicated by the index structure INX. That is, when objects move and their physical references change, the index structure INX is updated with the new physical reference such that future users will also find the customer object.

Page 25--delete the paragraph that carries over to page 26 and add a new paragraph after the first full paragraph (line 25)

As shown in FIG. 5b, the record storage contains a first (resident) part with the file descriptive attributes (each for example 32 bits) in the file descriptive part FDP. The file content, i.e. the data object itself ($\approx$10 kB) is stored in the disc part. One of the file descriptive attributes FD1 always points to the respective storage location of the file content in the disc part. When there is

a movement of the file content part, i.e. the object, according to the arrow M, then it is this file descriptive part FD1 that is updated with the new location of the file content in the disc part. The first part (resident part) can either be referenced by an index structure INX as illustrated in FIG. 5b or can be part of the index structure INX (i.e. the file descriptive part FDP) located in the right-hand column p_ID of the index structure INX. Alternatively, although not shown in FIG. 5b, another possibility is that the first part FDP (FD1) is directly referenced through a reference in the key, i.e. in the logical reference c_ID1. In all cases the file content can be moved as long as references in the first (resident) part are updated.

Page 26--amend the paragraph that carries over to page 27

[0081] However, according the ~~third~~ fourth embodiment, it is also possible to define the respective access frequency of a data object "relative" to the access frequency of another object. One possibility to realize such a "relative" access frequency determination is by using a doubly linked list and moving objects to the top of the list when they are accessed. For example, a particular page cache section PCS-j in FIG. 2 may have assigned to it a particular range of access frequencies, for example 5 to 10. Obviously, in this region two data objects having an access frequency of for example 6 and 9 may be located in this region. With respect to a third data object residing in the page cache section pch-J having an access frequency of 15, the first and second data object have relative "access frequency distances" of 9 and 6, respectively. Then, the data object ~~which~~ that has the closest "access frequency distance" will be moved to the next higher (hotter) page cache section. Thus, the moving of data objects can be made dependent on the relative access frequency. Of course, the associated list must contain the relative access frequency to all data objects respectively residing in the page cache section and/or the resident data section. Furthermore, it should be noted that also the relative access frequency (list) can be stored together with the data in the data object.

Page 27--amend the first full paragraph

[0082] As explained above, the present invention is particularly useful when used in connection with a data base system DBS comprising a primary disc memory D on which the data of the data base is stored and a data processing device [[DTD]] DPD.